Amendments to the Specification

Please replace paragraph starting on p. 2, line 17, with the following amended paragraph:

For example, during development of an integrated circuit, it is common to make changes to a netlist in order to add test circuits for testing the associated integrated circuit described by the netlist. Later in the development process, it may be desirable to apply the same changes to a revised netlist, which may be an altered, or updated, version of the netlist. According to conventional approaches, in this situation, the revised netlist is manually edited to include the same changes originally made to the netlist. This manual editing of the revised netlist may be cumbersome, error-prone, and time-consuming.

Please replace paragraph starting on p. 10, line 24, and ending on p. 11, line 9, with the following amended paragraph:

The lexical analyzer 302 is a component of the netlist compiler 206 ~~compiler~~ that reads the netlist 202 and the changes module 204 and produces, as an output, a set of tokens (not shown) that the parser 304 uses for syntax analysis. In general, lexical analysis involves breaking the HDL text into distinct, non-overlapping text strings in accordance with the rules of a specified language, such as Verilog or VHDL. These non-overlapping text strings are commonly referred to as "tokens." The lexical analyzer 302 may characterize some of these text strings where the text matches a keyword, or is a number or a symbol such as "&". In one embodiment, the lexical analyzer 302 comprises a conventional lexical analyzer, modified to recognize one or more of the HDL extensions found in the changes module 204 (FIG. 2) and described above.